Card Transactions Assignment

Diego Alonso

Overview

- Questions to answer
  - Top 10 most profitable customers by profile address (write an SQL query)
  - Segment current user base
  - Biggest cost line for the product
  - Recommendation based on your analysis
  - Forecast the average 12 months lifetime value for the user base
- Appendix. Understanding the data

Questions to answer

Top 10 most profitable customers by profile address country (write an SQL query)

```sql
WITH profitable_customers AS (
  SELECT
    profile_address_country,
    profile_owner_card,
    customer_total_profitability_in_gbp,
    ROW_NUMBER() OVER (PARTITION BY profile_address_country ORDER BY customer_total_profitability_in_gbp DESC) AS customer_index_by_country
  FROM `analysis_cards`
  WHERE customer_total_profitability_in_gbp > 0
)
SELECT
  *
FROM profitable_customers
WHERE customer_index_by_country <= 10
ORDER BY profile_address_country, customer_index_by_country
```


Please see the attached section **Understanding the data** at the end of this document where I explain how I aggregated the `transactions` and `cards` tables in order to produce the analysis table `` `analysis_cards` ``.

Note: For the rest of this analysis we'll assume a **one-to-one** `profile_owner_card` to `card_token` relationship to simplify our customer insights. There is only one instance (`profile_owner_card = 4965360`) where an owner has two card tokens. These card tokens were created at the same time and one was never used.

Top 10 most profitable customers by profile address country

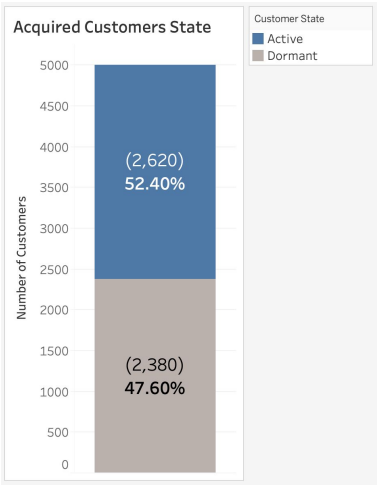
profile_address_country	profile_owner_card	customer_total_profitability_in_gbp	customer_index_by_country
AT	4439796	3.58	1
AT	1568231	2.47	2
AT	4436414	0.32	3
AT	2653092	0.05	4
BE	4950602	13.85	1
BE	94176	3.19	2
BE	2739529	2.46	3
BE	376992	0.66	4
BE	5150931	0.58	5
BG	5135106	0.2	1
CH	5274717	20.16	1
CH	189120	19.65	2
CH	809189	18.49	3
CH	1852364	13.15	4
CH	5007787	11.94	5
CH	4837940	11.12	6
CH	4927666	11.12	7
CH	1560328	11.07	8
CH	2333412	9.9	9
CH	4958716	9.2	10

The previous SQL query will only return profitable customers. As we can see, some profile address countries have **less than 10 profitable customers**.

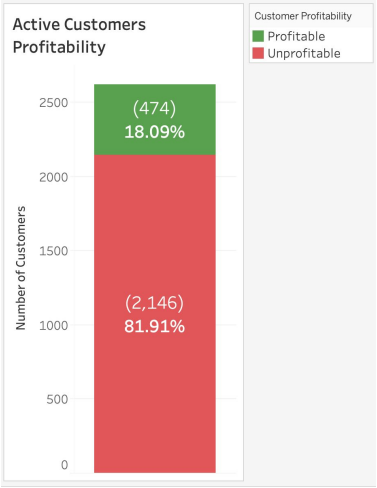
Segment current user base

We have **acquired** 5,000 customers who own cards.

A customer becomes **active** after making at least one transaction with their card.



Profitability is calculated aggregating the *revenue* and *cost* associated to every customer transaction.



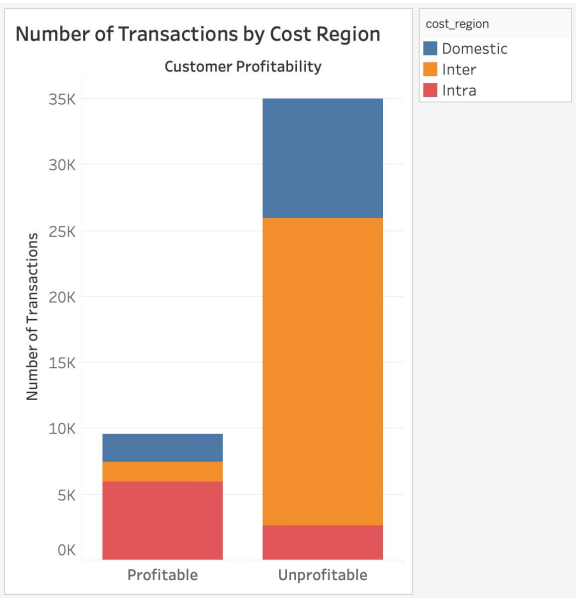
5

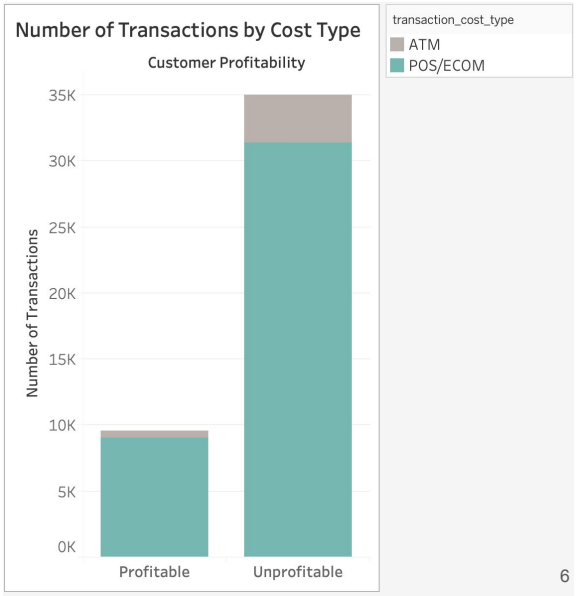
The **main goal for this task** is to understand in what segments we should focus our marketing efforts and pricing strategies.

First let's understand how many of our acquired users are active and then first try to start segmenting them based on profitability. Understanding **what makes a customer profitable** will be key to recommend how to **focus our marketing and pricing strategies**.

As a reminder the attached section **Understanding the data** at the end of this document explains how I've calculated profitability for each customer aggregating their transactions against the `cost_line` *fees* on a *currency neutral (GBP)* basis.

Customer Profitability Segments





6

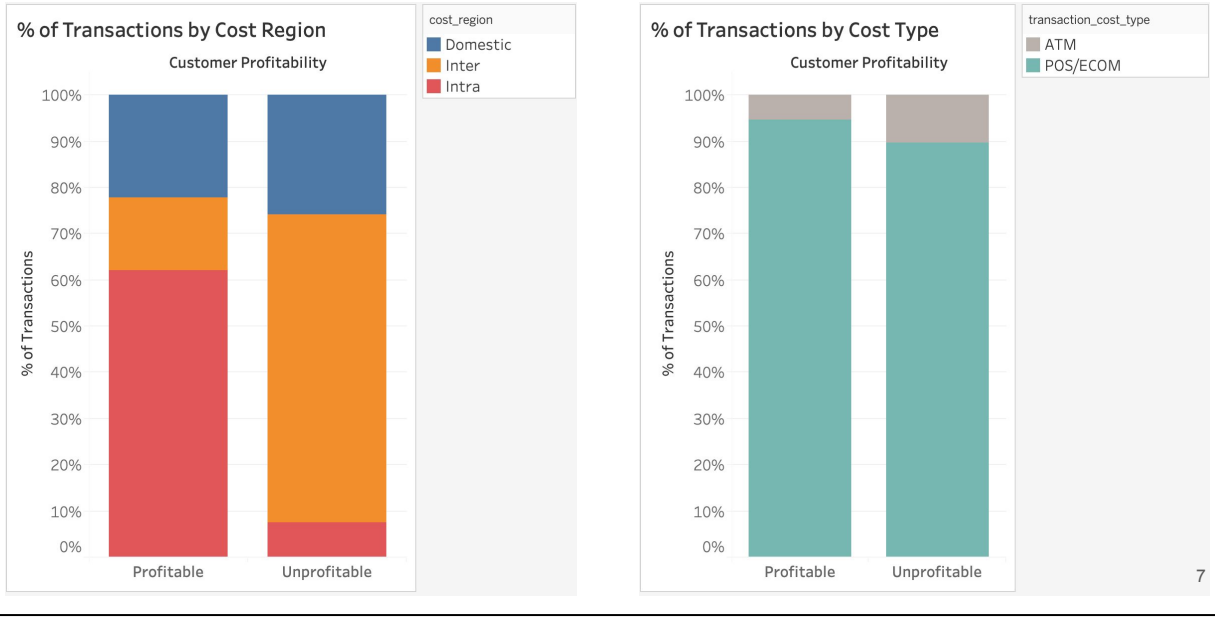
I have divided our customer base in *two segments*: **Profitable** and **Unprofitable customers**.

Profitability is mostly determined by how much we are taxed by the cost structure **cost line fees**. So let's see how customer profitability is distributed by **cost region** and **transaction cost type**.

The total number of transactions made by unprofitable customers account for 3 times more than profitable customer ones.

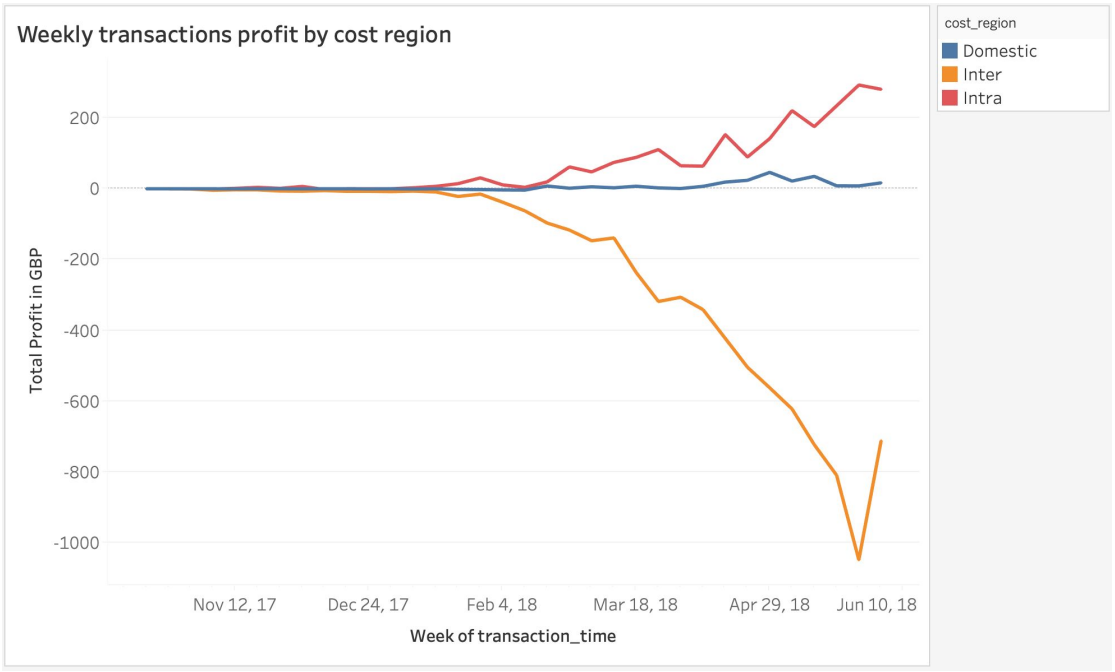
We can also see how **profitable customers** made more *Intra cost region* transactions. While **unprofitable customers** made more *Inter cost region* transactions and *ATM cost type* transactions.

Customer Profitability Segments

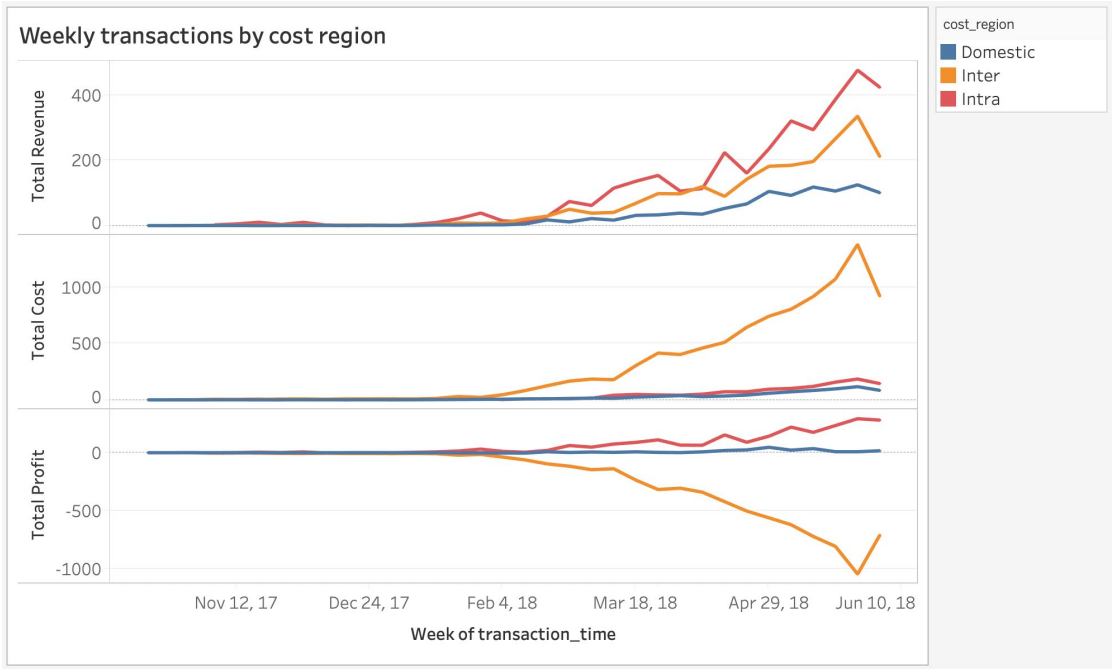


When observing the percentage distribution of transactions by **cost type** we see that **unprofitable customers** made proportionally a bit more transactions by **ATM** than profitable customer ones. This doesn't come as a surprise since the *intercharge fee* is negative for ATM transactions. Nevertheless there a few situations where the conversion revenue could make an ATM transaction profitable.

Something more noticable is that when analyzing **cost region** we see that most of the transactions made by our **profitable customers** happened on the **Intra cost region** and **unprofitable customers** made more transactions on the **Inter cost region**. This is directly correlated to cost structure as we see on our database that the **Inter region has the highest cost line fees** of the three regions.

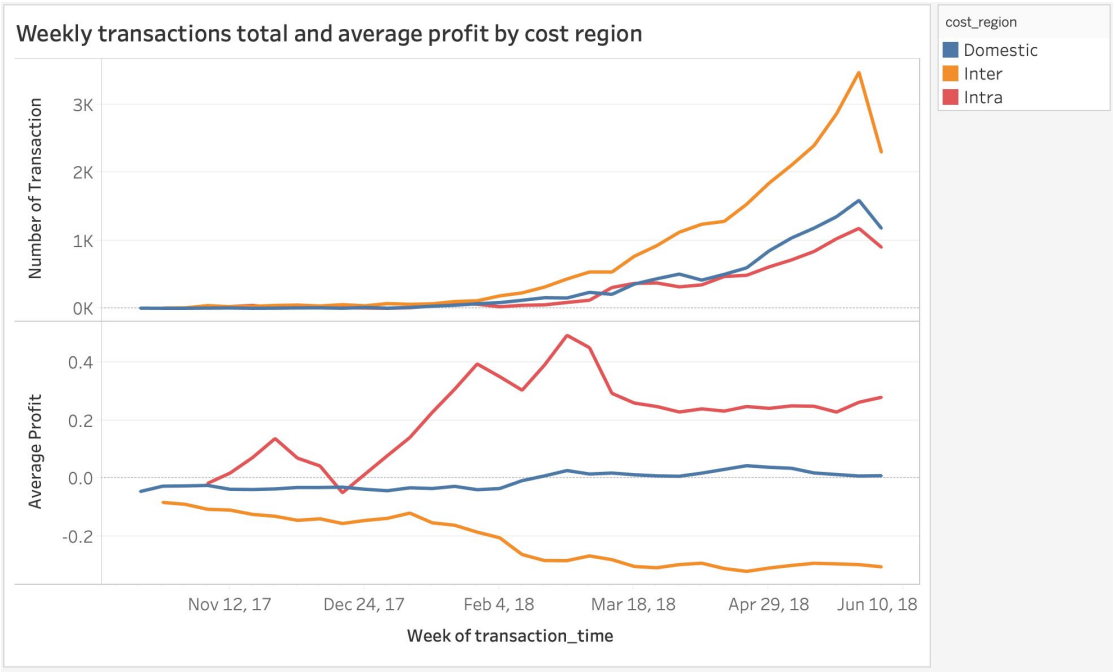


When we aggregate transactions profit on a weekly basis we clearly see how the **Inter cost region is just not profitable at all.**



9

Even though the **Inter cost region** does generate revenue, its aggregate cost is significantly higher and considerably lower than the revenue generated by the **Intra cost region** (who has less transactions as we'll see next).



As we mentioned on the previous slide here we can see how the **total number of transactions from the Inter cost region** surpasses for almost 3 times the total transactions of the **Intra** and **Domestic** regions. Even though *Inter* is the most costly region, it is the one with the most engagement.

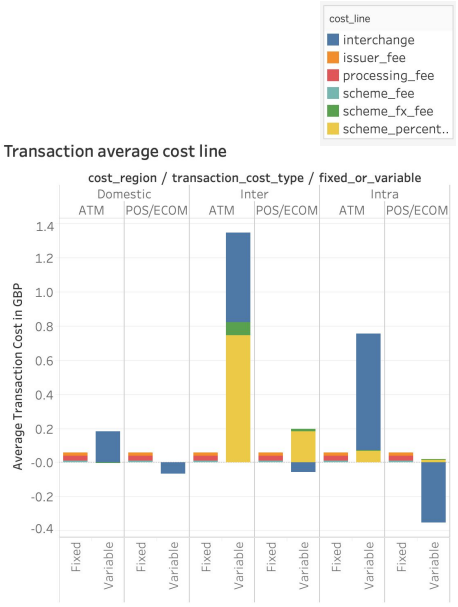
When analyzing the **weekly transactions average profit** we can see that we usually break even with the *Domestic* cost region. Ideally if we could bring down the cost from the *Inter* region and increase the number of transaction of the *Intra* region **we could break even for the whole product!!**

Biggest cost line for the product

Transaction average cost line table

cost_region	fixed_or_variable	cost_line	transaction_cost_type ATM	POS/ECOM
Domestic	Fixed	issuer_fee	0.020	0.020
		processing_fee	0.030	0.030
		scheme_fee	0.010	0.010
	Variable	interchange	0.186	-0.065
		scheme_fx_fee	0.000	0.000
		scheme_percentage_fee	0.000	0.000
Inter	Fixed	issuer_fee	0.020	0.020
		processing_fee	0.030	0.030
		scheme_fee	0.010	0.010
	Variable	interchange	0.528	-0.056
		scheme_fx_fee	0.075	0.013
		scheme_percentage_fee	0.747	0.186
Intra	Fixed	issuer_fee	0.020	0.020
		processing_fee	0.030	0.030
		scheme_fee	0.010	0.010
	Variable	interchange	0.683	-0.352
		scheme_fx_fee	0.000	0.000
		scheme_percentage_fee	0.072	0.022

Transaction average cost line

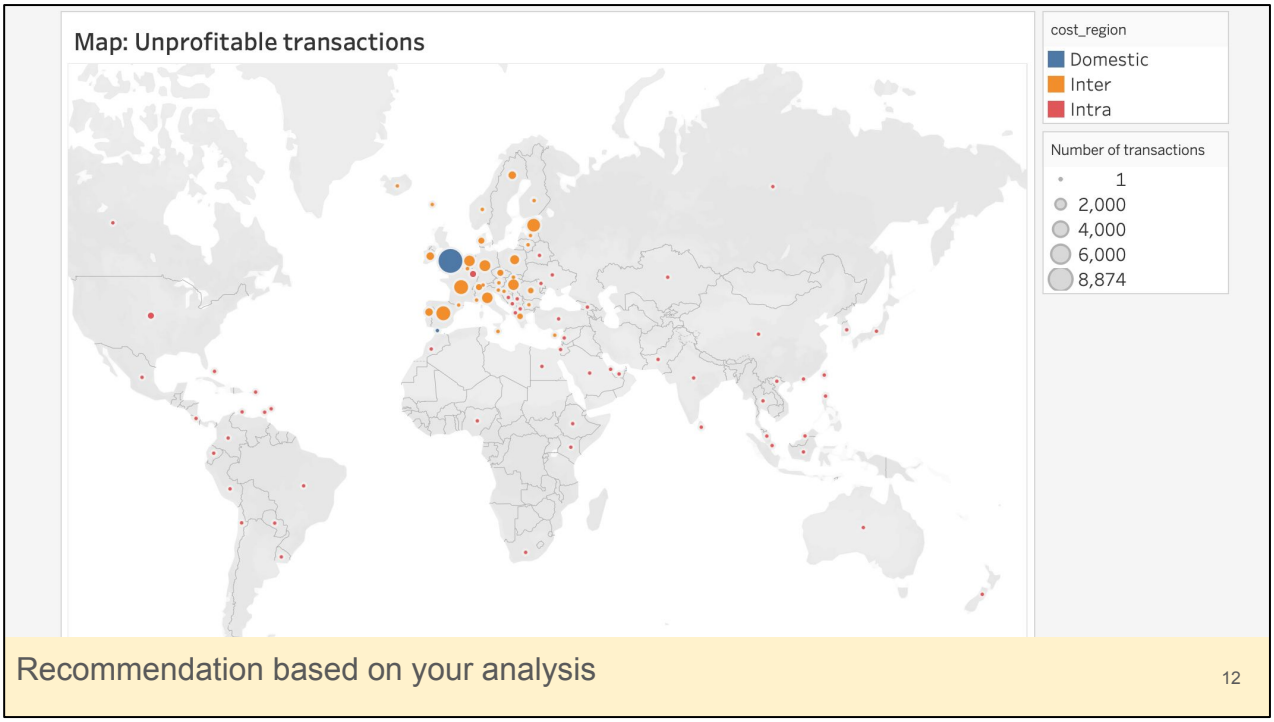


11

As we saw on the previous slide the biggest cost line resides on the **Inter region**.

More specifically if we break down the regions cost lines and transaction types we can see that **schema_percentage_fee** is the main driver of cost on the *Inter* region. **Operationally I would recommend** to reevaluate how can we renegotiate this fee or look for ways to avoid it if possible by implementing certain stage of the cost line process in-house.

Note/Caveat: I used the converted intercharge amount originally provided on the `transactions` table throughout our analysis. This amount only matched on the *Domestic* region when recalculating with the rate provided on the `cost structure` table.

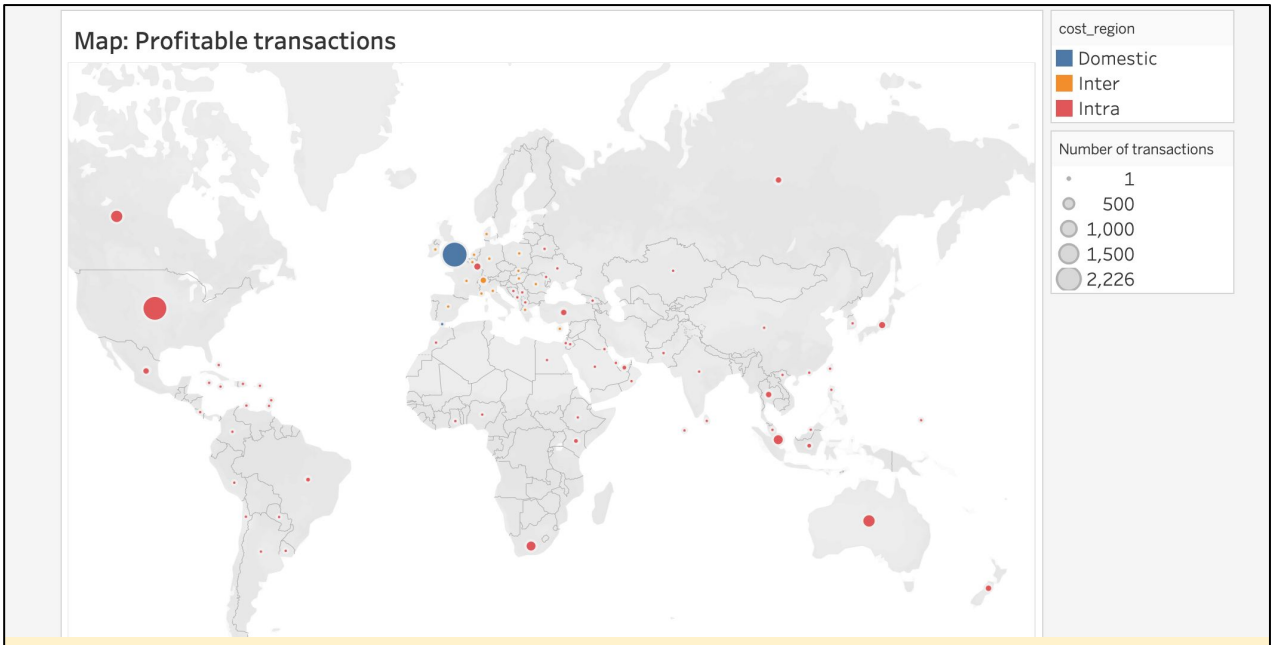


Recommendation based on your analysis

Besides looking for operational ways to reduce the cost line fees as I've suggested before in order to drive us to profitability on the *Inter* region.

I'd also recommend to the Marketing team to keep targeting users that spend on the *Intra* region. There seems still to be opportunity to acquire new users on the *Intra* region and also remind existing (and even dormant users) in this region about their card.

Let's **compare the number of Unprofitable vs. Profitable transactions** using the merchant address country colored by cost region.



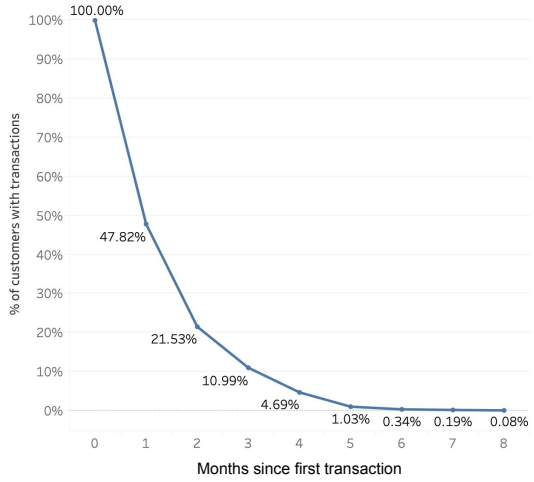
Recommendation based on your analysis

After seeing the *Intra* countries that are driving the highest number of profitable transactions, intuitively I'd recommend the Marketing team to focus their effort of targeting customers showing behaviours of transacting in the following countries: *United States, Canada, Australia, Singapore, and South Africa*.

Forecast the average 12 months lifetime value for the user base.

Customer Retention Analysis

Retention by month since first transaction



Customer Growth Cohort Analysis

First transaction month cohorts

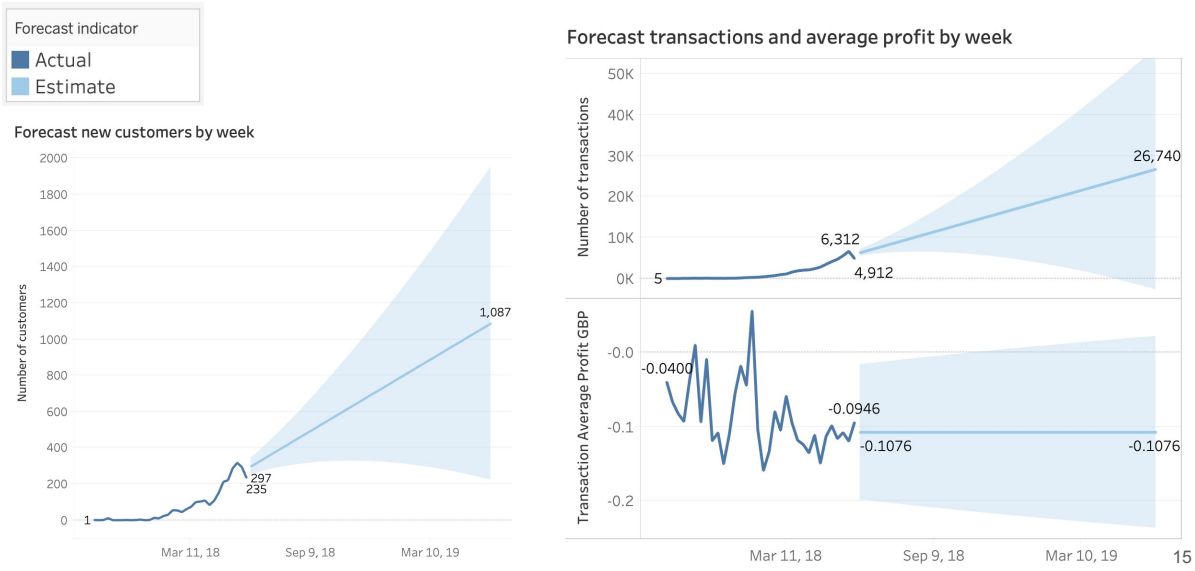
Month of Cohort	October 2017	November 2017	December 2017	January 2018	February 2018	March 2018	April 2018	May 2018	June 2018
October ..	2	1	2	1	1	2	1	1	2
Novemb..		11	7	9	7	7	7	6	4
Decembe..			5	4	4	3	4	4	2
January ..				35	22	20	20	23	14
February..					174	132	127	123	88
March 2..						360	236	224	134
April 2018							485	326	178
May 2018								1,191	525
June 2018									357

14

Before trying to forecast 12-month CLV, let's understand how our customer base is growing and what's our **retention rate**.

We don't have enough data to make any hard claims on retention rate over the long run and by looking at the cohorts it seems that the product started getting traction on the last 4 months.

Forecast the average 12 months lifetime value for the user base.



I opted to use Tableau's exponential smoothing technique in order to forecast the 12-month number of customers, number of transactions and average profit per transaction.

Forecast the average 12 months lifetime value for the user base.

$$CLV = Annual\ Revenue\ Per\ User \div Churn\ Rate$$
$$= (Total\ transactions \times Average\ profit\ per\ transaction \div Total\ customers) \div Churn\ Rate$$

Actual + estimated **total customers** = 39,265

Actual + estimated **total transactions** = 923,689

Estimated **average profit per transaction =** -0.10 GBP

Churn rate = 52%

12-month Forecast LTV = -4.86 GBP

16

I used a simplified CLV function calculated with ARPU over Churn Rate.

The actual and estimated customers and transactions are calculated with the running total from the weekly growth forecasted, see the Understanding the data section for the reference table.

I arbitrarily chose the first month **churn rate** based on our previous retention analysis, my intuition tells me that the yearly churn rate will range between 52% and 80%.

If we were to multiply the estimated CLV for the number of estimated customers we'll see the overall value of the product would be: $-190{,}828$ GBP. As discussed before any small restructuring on cost line structure could bring great impact to the overall product profitability.

Thanks for reading

Don’t forget to review the following section

Understanding the data

Creating aggregate tables

Table name: **`analysis_transactions`**

In order to facilitate our analysis I've created an aggregate table based on our original **`transactions`** table, adding **currency neutral** (GBP) **billing amount** and **interchange** fees.

```
billing_amount_in_gbp: NUMERIC,
inter_change_fee_in_gbp: NUMERIC
```

In addition I've calculated and added the cost structure **fixed and variable scheme fees** in GBP.

```
scheme_fee: NUMERIC,
issuer_fee: NUMERIC,
processing_fee: NUMERIC,
scheme_percentage_fee: NUMERIC,
scheme_fx_fee: NUMERIC,
test_interchange_fee: NUMERIC
```

And finally aggregated the **revenue**, **cost** and final **profit** for each transaction based on the *schema fees* (including *interchange*) and *conversion revenue*.

```
total_revenue: NUMERIC,
total_cost: NUMERIC,
total_profit: NUMERIC
```

- **Note**: We use `inter_change_fee_in_gbp` calculated from the original **`transactions`** table throughout the analysis, the recalculated `test_interchange_fee` is just for the sake of completion.

```
WITH transactions_in_gbp AS (
  SELECT
    t.*,
    ROUND(t.billing_amount_value * r.rate, 2) AS billing_amount_in_gbp,
    ROUND(t.inter_change_fee * r.rate, 2) AS inter_change_fee_in_gbp
  FROM `transactions` t
  LEFT JOIN `rates` r
    ON r.code = t.billing_amount_currency
),
transactions_in_gbp_cost_structure AS (
  SELECT
    t.*,
```

```
    c.cost_line,
    c.fixed_or_variable,
    -- Schema FX fee should only apply when there's currency exchange between original
transaction amount and billed amount
    (CASE WHEN c.cost_line = 'scheme_fx_fee' AND t.amount_currency = t.billing_amount_currency
THEN 0
      WHEN c.fixed_or_variable = 'Fixed' THEN ROUND(c.cost_in_gbp, 2)
      WHEN c.fixed_or_variable = 'Variable' THEN ROUND(c.variable_fee * t.billing_amount_in_gbp,
2) END
    ) AS fixed_or_variable_cost_in_gbp
  FROM transactions_in_gbp t
  JOIN `cost_structure` c
    ON c.transaction_cost_type = t.transaction_cost_type AND c.cost_region = t.cost_region
),
transactions_in_gbp_cost_structure_pivot AS (
  SELECT
    id,
    SUM(CASE WHEN cost_line = 'scheme_fee' THEN fixed_or_variable_cost_in_gbp END) AS scheme_fee,
    SUM(CASE WHEN cost_line = 'issuer_fee' THEN fixed_or_variable_cost_in_gbp END) AS issuer_fee,
    SUM(CASE WHEN cost_line = 'processing_fee' THEN fixed_or_variable_cost_in_gbp END) AS
processing_fee,
    SUM(CASE WHEN cost_line = 'scheme_percentage_fee' THEN fixed_or_variable_cost_in_gbp END) AS
scheme_percentage_fee,
    SUM(CASE WHEN cost_line = 'scheme_fx_fee' THEN fixed_or_variable_cost_in_gbp END) AS
scheme_fx_fee,
    SUM(CASE WHEN cost_line = 'interchange' THEN fixed_or_variable_cost_in_gbp END) AS
test_interchange_fee
  FROM transactions_in_gbp_cost_structure
  GROUP BY 1
)
SELECT
  *,
  -- Total revenue = inter_change_fee_in_gbp (if positive) + conversion_revenue
  ROUND(CASE WHEN inter_change_fee_in_gbp > 0 THEN inter_change_fee_in_gbp + conversion_revenue
    ELSE conversion_revenue END, 2) AS total_revenue,

  -- Total cost = inter_change_fee_in_gbp (if intercharge) + schema fixed and variable fees
  ROUND(CASE WHEN inter_change_fee_in_gbp < 0 THEN (inter_change_fee_in_gbp * -1) + scheme_fee +
issuer_fee + processing_fee + scheme_percentage_fee + scheme_fx_fee
    ELSE scheme_fee + issuer_fee + processing_fee + scheme_percentage_fee + scheme_fx_fee END, 2)
AS total_cost,

  ROUND(conversion_revenue + inter_change_fee_in_gbp - scheme_fee - issuer_fee - processing_fee -
scheme_percentage_fee - scheme_fx_fee, 2) AS total_profit
FROM transactions_in_gbp t
JOIN transactions_in_gbp_cost_structure_pivot c
  USING(id)
```

analysis_transactions	Field names	Type
	id	INTEGER
	transaction_time	TIMESTAMP
	card_token	STRING

merchant_address_country	STRING
cost_region	STRING
transaction_type	STRING
transaction_cost_type	STRING
transaction_personal_details_input	STRING
transaction_card_ownership_proof_method	STRING
state	STRING
decline_reason	STRING
amount_currency	STRING
amount_value	NUMERIC
billing_amount_currency	STRING
billing_amount_value	NUMERIC
interchange_currency	STRING
inter_change_fee	NUMERIC
conversion_revenue_currency	STRING
conversion_revenue	NUMERIC
merchant_name	STRING
category	STRING
billing_amount_in_gbp	**NUMERIC**
inter_change_fee_in_gbp	**NUMERIC**
scheme_fee	**NUMERIC**
issuer_fee	**NUMERIC**
processing_fee	**NUMERIC**
scheme_percentage_fee	**NUMERIC**
scheme_fx_fee	**NUMERIC**
test_interchange_fee	**NUMERIC**
total_revenue	**NUMERIC**
total_cost	**NUMERIC**

total_profit	**NUMERIC**

Table name: **analysis_cards**

In order to make easier customer analysis I've created an aggregate table based on our original **cards** table and new **analysis_transactions** table and added the following new metrics:

Engagement

```
first_transaction: TIMESTAMP,
last_transaction: TIMESTAMP,
total_billing_abount_in_gbp: NUMERIC,
average_billing_amount_in_gbp: NUMERIC,
number_of_transactions: INTEGER,
number_of_authorized: INTEGER,
number_of_declined: INTEGER,
```

Total customer profitability KPI

```
customer_total_profitability_in_gbp: NUMERIC,
```

Profitability by transactions cost region

```
profitability_domestic: NUMERIC,
profitability_inter: NUMERIC,
profitability_intra: NUMERIC,
number_of_domestic: INTEGER,
number_of_inter: INTEGER,
number_of_intra: INTEGER,
```

Profitability by transactions cost type

```
profitability_atm: NUMERIC,
profitability_pos_ecom: NUMERIC
number_of_atm: INTEGER,
number_of_pos_ecom: INTEGER,
```

```
WITH card_stats AS (
  SELECT
    card_token,
    MIN(t.transaction_time) AS first_transaction,
    MAX(t.transaction_time) AS last_transaction,
    SUM(t.billing_amount_in_gbp) AS total_billing_abount_in_gbp,
    ROUND(AVG(t.billing_amount_in_gbp), 2) AS average_billing_amount_in_gbp,
    COUNT(t.id) AS number_of_transactions,
    SUM(t.total_profit) AS customer_total_profitability_in_gbp,
    SUM(CASE WHEN t.cost_region='Domestic' THEN 1 END) AS number_of_domestic,
    SUM(CASE WHEN t.cost_region='Inter' THEN 1 END) AS number_of_inter,
    SUM(CASE WHEN t.cost_region='Intra' THEN 1 END) AS number_of_intra,
    SUM(CASE WHEN t.transaction_cost_type='ATM' THEN 1 END) AS number_of_atm,
```

```
    SUM(CASE WHEN t.transaction_cost_type='POS/ECOM' THEN 1 END) AS number_of_pos_ecom,
    SUM(CASE WHEN t.state='SUCCESS' THEN 1 END) AS number_of_authorized,
    SUM(CASE WHEN t.state='FAIL' THEN 1 END) AS number_of_declined,
    SUM(CASE WHEN t.cost_region='Domestic' THEN t.total_profit END) AS profitability_domestic,
    SUM(CASE WHEN t.cost_region='Inter' THEN t.total_profit END) AS profitability_inter,
    SUM(CASE WHEN t.cost_region='Intra' THEN t.total_profit END) AS profitability_intra,
    SUM(CASE WHEN t.transaction_cost_type='ATM' THEN t.total_profit END) AS profitability_atm,
    SUM(CASE WHEN t.transaction_cost_type='POS/ECOM' THEN t.total_profit END) AS
profitability_pos_ecom
  FROM `cards` c
  LEFT JOIN `analysis_transactions` t
    USING(card_token)
  GROUP BY 1
)
SELECT
  *
FROM `cards`
JOIN card_stats USING(card_token)
```

analysis_cards **Field names**	**Type**
card_token	STRING
card_produced_time	TIMESTAMP
profile_owner_card	INTEGER
is_active	BOOLEAN
profile_address_country	STRING
age_years	INTEGER
email_domain	STRING
card_delivery_country	STRING
card_delivery_city	STRING
first_transaction	**TIMESTAMP**
last_transaction	**TIMESTAMP**
total_billing_abount_in_gbp	**NUMERIC**
average_billing_amount_in_gbp	**NUMERIC**
number_of_transactions	**INTEGER**
customer_total_profitability_in_gbp	**NUMERIC**
number_of_domestic	**INTEGER**
number_of_inter	**INTEGER**
number_of_intra	**INTEGER**

number_of_atm	INTEGER
number_of_pos_ecom	INTEGER
number_of_authorized	INTEGER
number_of_declined	INTEGER
profitability_domestic	NUMERIC
profitability_inter	NUMERIC
profitability_intra	NUMERIC
profitability_atm	NUMERIC
profitability_pos_ecom	NUMERIC